USAISEC
US Army Information Systems Engineering Command
Fort Huachuca, AZ 85613-5300

U.S. ARMY INSTITUTE FOR RESEARCH
IN MANAGEMENT INFORMATION,
COMMUNICATIONS, AND COMPUTER SCIENCES

The Software Support Qualitative Assessment Methodology

Volume I

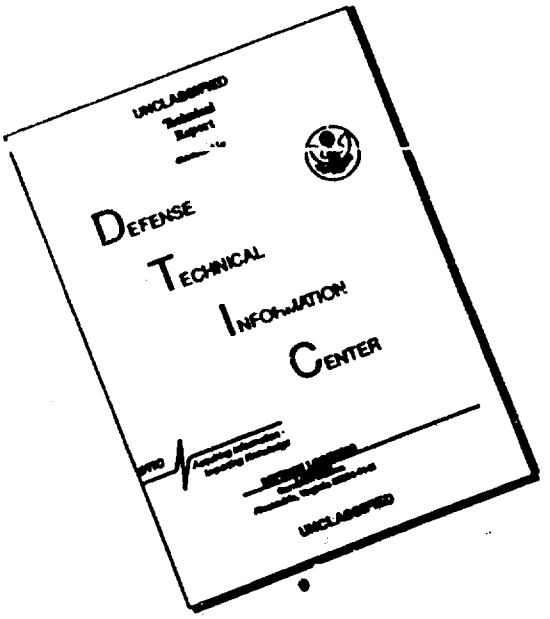
Developing Quality Measures for Information Systems Support

March 1991

ASQB-GI-91-016

AIRMICS
115 O'Keefe Building
Georgia Institute of Technology
Atlanta, GA 30332-0800

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704--188
Exp. Date: Jun 30, 1986

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS NONE
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION / AVAILABILITY OF REPORT N/A
2b. DECLASSIFICATION / DOUWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) ASQB-GI-91--016	5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A
6a. NAME OF PERFORMING ORGANIZATION The Center for Information Management Research 6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION N/A
6c. ADDRESS (City, State, and ZIP Code) SERC Georgia Institute of Technology Atlanta, GA 30332 Dept of MIS University of Arizona Tucson, AZ 85613	7b. ADDRESS (City, State, and Zip Code) N/A
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AIRMICS 8b. OFFICE SYMBOL (if applicable) ASQB - GI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code) 115 O'Keefe Bldg., Georgia Institute of Technology Atlanta, GA 30332-0800	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 62783A; PROJECT NO. DY10; TASK NO. 02-01-01; WORK UNIT ACCESSION NO

11. TITLE (Include Security Classification) The Software Support Qualitative Assessment Methodology
Volume I
Developing Qualitative Measures for Information Systems Support
(UNCLASSIFIED)

12. PERSONAL AUTHOR(S)
W. Michael McCracken, Elizabeth Mynatt, Christopher Smith (GIT)
J.F. Nunamaker, Ai-Mei Chang, Titus Purdin, Richard Orwig, Amit Vyas (Univ of Arizona)

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
final report	FROM _____ TO _____	1991, March, 22	44

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	life cycle metrics; systems support; support measures; qualitative assessment; supportability measures; assessment measures; readiness measures; information systems; software maintenance; support tools; tools management;

19. ABSTRACT (Continue on reverse if necessary and identity by block number)

The Software Supportability Qualitative Assessment Methodology is a five volume reference set that provides measures to aid in the support of information systems. The volumes are aimed at improving the support process by more accurately assessing the capabilities of support organizations, qualitatively measuring the supportability of fielded systems and evaluating the operational readiness of fielded systems. The five volumes are:

I. Developing Quality Measures for Information Systems Support
II. The Review of Metrics for Developing an Information Systems Support Measurement Framework
III. Implementing the Software Supportability Measure
IV. Implementing the Support Organization Assessment Measure
V. Implementing the Operational Readiness Measure

This volume describes the three measures along with the model of information systems support that the measures are designed to satisfy. It is the main volume and should be consulted before implementing the other volumes. Topics discussed in volume I are described in more detail in the other volumes.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
[X] UNCLASSIFIED / UNLIMITED [ ] SAME AS RPT [ ] DTIC USERS	UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Howard C "Butch" Higley	22b. TELEPHONE (Include Area Code) (404, 894-3110	22c. OFFICE SYMBOL ASQB-GI

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

The research herein was performed for the Army Institute for Research in Management Information, Communications, and Computer Sciences (AIRMICS), the RDTE organization of the U.S. Army Information Systems Engineering Command (USAISEC). The sponsor for the project was the Office of the Director of Information Systems for Command, Control, Communications, and Computers (ODISC4). The principal investigator was from The Center for Information Management Research (CIMR), W. Michael McCracken of the Georgia Institute of Technology.

This research report is not to be construed as an official Army position, unless so designated by other authorized documents. Material included herein is approved for public release, distribution unlimited, and is not protected by copyright laws. Your comments on all aspects of the document are solicited.

THIS REPORT HAS BEEN REVIEWED AND IS APPROVED

s/ _____
Glenn E. Racine
Chief
CISD

s/ _____
John R. Mitchell
Director
AIRMICS

The Software Support
Qualitative Assessment Methodology

Volume I

Developing Quality Measures for Information Systems Support

Prepared by

The Center for Information Management Research

for the

U.S. Army Institute for Research in Management

Information, Communications, and Computer Science

(AIRMICS)

Contract No. ECD-8904815

W. Michael McCracken, Elizabeth Mynatt, Christopher Smith

Software Engineering Research Center

Georgia Institute of Technology

J. F. Nunamaker Jr., Ai-Mei Chang, Titus Purdin

Richard Orwig, Amit Vyas

Department of Management Information Systems

University of Arizona

December 1990

The **Software Supportability Qualitative Assessment Methodology** is a five volume reference set that provides measures to aid in the support of information systems. These manuals are aimed at improving the support process by more accurately assessing the capabilities of support organizations, quantitatively measuring the supportability of fielded systems and evaluating the operational readiness of fielded systems.

Volume I, *Developing Quality Measures for Information Systems Support*, describes the three measures along with the model of information system support that the measures are designed to satisfy. This is the main volume of the set and should be consulted before implementing the measures described in more detail in the other volumes.

Volume II, *The Review of Metrics for Developing an Information Systems Support Measurement Framework*, provides a survey and evaluation of current metrics in terms of information systems support. Specifically, three classes of metrics are reviewed: software product metrics, life cycle process metrics, and process management metrics.

Volume III, *Implementing the Software Supportability Measure*, provides instructions for collecting data for the measure, compiling the measure by evaluating the data, and interpreting the final result. The volume also contains guidelines for improving the supportabilty of an information system based on its evaluation. Specifically, the volume contains resource estimations for compiling and evaluating the measure, questionnaires for collecting the required data and step-by-step instructions for measuring the supportability of an information system.

Volume IV, *Implementing the Support Organization Assessment Measure*, provides instructions for collecting data for the assessment, conducting the assessment, and interpreting the final result. The volume also contains guidelines for improving the capabilities of a support organization based on its evaluation. Specifically, the volume contains resource estimations for conducting and evaluating the assessment, questionnaires for collecting the required data and step-by-step instructions for measuring the capabilities of a support organization.

Volume V, *Implementing the Operational Readiness Measure*, provides instructions for collecting data for the measure, compiling the measure by evaluating the data, and interpreting the final result. The volume also contains guidelines for improving the operational readiness of an information system based on its evaluation. Specifically, the volume contains resource estimations for compiling and evaluating the measure, questionnaires for collecting the required data and step-by-step instructions for measuring the operational readiness of an information system.

Contents

1 Executive Summary

The Software Support Qualitative Assessment Methodology (contract no. ECD-8904815) is a methodology for developing and implementing a comprehensive framework of support measures for use by U. S. Army information systems support organizations. The support measures allow an information systems support organization to evaluate its effectiveness of information systems support, the supportability of their fielded information systems, and the operational readiness of the information systems. The Center for Information Management Research (CIMR) at the Georgia Institute of Technology and the University of Arizona has developed these measures. In addition, we have developed a set of guidelines for a support organization to implement a support measurement program.

The motivation for developing this methodology arises from the fact that the support of software now consumes an increasing majority of total life cycle cost [SB88]. Because information systems are typically long-lived, the support organization must be able to respond effectively to the arising software problems, a changing of information system requirements, and a changing user population. A Software Support Qualitative Assessment Methodology allows the support organization to understand and improve their support process, which, in turn, allows it to effectively respond to the above problems.

The following paragraphs outline the measures comprising our developed methodology, an overview of the information provided in this five-volume document, and an overview of the research activities conducted during the course of this project.

Support Measures

The goal of the Software Support Qualitative Assessment Methodology is the development of a comprehensive set of support measures, which take into account differing perspectives within the information systems support environment. Within the support environment, there is the support organization, the information system(s), and the users. Depending upon one's perspective (support organization management, support technician, or user), certain measures may be more useful in interpreting the capability to adequately support given information system(s). We propose three measures to accommodate these varying viewpoints – when taken together, provide a comprehensive view of the state of information systems support. The three measures are:

- Supportability
- Support Organization Assessment
- Operational Readiness

These measures are briefly described in the following paragraphs.

Supportability

The **supportability** of an information system is the measure of the adequacy of products, resources, and procedures to facilitate:

- The intended operation of the software system or the restoration of the system to its intended state; and
- The modification of the software system to meet new requirements.

Supportability takes into account the point of view of those directly maintaining an information system. It is intended to answer such questions as "Is the information system maintainable?" and, "Are the resources and procedures specifically used to support this information system adequate?"

The Supportability measure is comprised of software maintainability, support management, and support resource metrics. Our measure is essentially risk-based, with "risk" being defined as the possibility that user expectations for the given information system will not be met (caused by software failures, inability to meet new requirements, etc.). The measure is also intended to aid in identifying components significantly producing any increased support risk.

Support Organization Assessment

A **support organization assessment measure** is a measure utilized by the information systems support organization to determine the effectiveness of policies, procedures, resource management, and personnel management in fulfilling the organization's support objectives. The assessment measure takes into account the perspective of those managing the support process and provides an overall view of support organization effectiveness.

The support organization assessment measure answers the question, "Can the support organization capably and adequately maintain its collection of information systems?" The value of the measure is the "level" of maturity of the support organization with respect to their support process. The levels of maturity are: Ad-Hoc, Repeatable, Methodology, Control and Optimal.

Operational Readiness

The **operational readiness** of a software system is the ability of the software system to effectively perform its intended function, based on:

- The correct operational state of the system;
- The reliability of the system; and
- The supportability of the system.

The operation readiness measure is designed for the users' perspective of information systems support. The measure addresses such questions as "Is the information system up and running when I need it?" and, "When I use the system, can I expect correct results?" The operational readiness measure is mainly predictive because we are usually interested in the operational state of an information system both at a given "present" time and for some immediate future time period.

Like supportability, operational readiness is a risk-based measure. The value of the measure is the probability that an information system will perform its intended function. In addition, we borrow terminology previously applied to military hardware equipment to interpret the measure. The terms used to denote the operational readiness are **red** (information system is in a serious state of disrepair), **yellow** (system is marginally operational), and **green** (system is fully operation and functioning without difficulty). The appropriate term can be assigned based on the computed value of the operational readiness measure.

Characteristics of Measures

The above three measures have been designed to incorporate two desired characteristics. First, the measures should be easy to compute. Ease of computation involves use of a simple data gathering method, gathering a minimal set of data, and using straightforward conversions from raw to derived measures.

Second, the measures should be easy to interpret. Ease of interpretation implies a presentation of the measurement in the language understood by the user. As indicated in the above discussion, we have chosen to present measures both as risk-based (supportability and operational readiness) and as based on an easily understandable level of abstraction (all three measures).

Research Results

The results of our research and development of the above measures are summarized in this five-volume document. This volume (Volume I) contains information about the background and objectives of our research, an exposition of the foundation of our proposed support measures, a description of the three measures, a cost/benefit analysis for implementing these measures, and possible areas of future research for which our study lays a foundation.

The other four volumes contain more detailed background and implementation information. Volume II contains a review of existing metrics applicable to the construction of our support measures and an outline of a model around which these metrics can be used to build the top-level support measures. Volume III contains information for implementing the Supportability measure. Volume IV contains information for implementing the Support Organization Assessment measure. Volume V contains information for implementing the Operational Readiness measure.

Project History

The research for the Software Support Qualitative Assessment Methodology was conducted during the sixteen-month period from September, 1989, through December, 1990. The research project consisted of four distinct phases:

- Review of existing support measures
- Development of information systems support model

- Collection of information systems support data
- Construction of support measures

Review of Existing Support Measures

During the first stage of the project, we conducted a review of existing life cycle metrics proposed in the literature which are applicable to the information systems support cycle. These metrics are outlined in Volume II of this document.

Development of Information Systems Support Model

The next stage of our research involved the development of an information systems support model serving as a foundation for the support measures we have developed. The model is described in Section 5 of this volume as well as in Volume II.

Collection of Information Systems Support Data

In the next stage of this research project, we conducted several on-site visits to various U. S. Army Information Systems Support facilities (see Appendix C of this volume for details). The purpose of these visits was to gather support organization and information systems data to help us construct accurate and realistic support measures.

Construction of Support Measures

From the review of existing metrics, a theoretical model for information systems support, and data gathered through visits to support organizations, we constructed the top-level support measures. The rationale for the given construction of the three top-level support measures (outlined in the first two volumes of this document) rests on the validity of our developed model of information systems support and the ability to realistically collect valid data in a support organization environment.

Areas for Further Research

The results of this research are the initial support qualitative methodology, including the three high-level support measures, and methodology implementation guidelines. Our study lays the foundation for additional studies to refine and validate the qualitative assessment methodology and for studies of reverse engineering, a process closely coupled with the support of information systems (see Section 10 of this volume).

The refinement and validation studies would focus on the thorough evaluation of the proposed construction of the support measures and subsequent refinement of the initially proposed methodology. In addition, studies of methodology usage, automated methodology assistance tools, and information systems users need to be conducted. The reverse engineering studies would emphasize the development of a reverse engineering decision model,

the development of a general reverse engineering methodology, and the analysis of reverse engineering cost and risk factors.

2 Motivation

A major contributor to the life cycle cost of information systems is the cost of supporting these systems. Not only does support cost now consume a majority of the total life cycle cost [SB88], there does not exist an effective means for determining support cost drivers and reducing support cost. The support process remains poorly understood, and there is no comprehensive set of quality measures for information systems support that may be utilized by those directly involved in supporting and using these systems. The information systems support organization must have a method of evaluating their capability to adequately support their collection of information systems. Additionally, users and supporters of information systems need measures of the supportability and operational readiness of those systems.

The Software Support Qualitative Assessment Methodology (contract no. ECD-8904815) is a methodology for developing and implementing a comprehensive framework of support measures for use by U. S. Army information systems support organizations. These measures give a support organization a method of evaluating their capability to adequately support their collection of information systems. The measures also allow the organization to determine the supportability and operational readiness of their fielded information systems. In addition to the measures themselves, guidelines for a support organization's incorporation of the measures have been developed. These measures and guidelines have been developed by the Center of Information Management Research (CIMR) at Georgia Institute of Technology and the University of Arizona.

In this document, we outline the background and objectives of this research and discuss the foundations of the proposed support measures. The background consists of a brief review of existing metrics for information systems support. The objectives outline the primary goals of this research, intended audience, and the characteristics of the support measures we have attempted to incorporate. And the foundations for three proposed support measures, supportability, the support organization assessment, and operational readiness, are examined. Also, we discuss the cost and benefits of implementing this methodology and possible areas of future research.

3 Survey of Existing Measures

Rather than "reinvent the wheel" and propose an entirely new class of metrics for information systems support, it would be much more preferable to develop a set of support measures by at least partially utilizing existing metrics. It is likely a framework of support measures can be developed as such. The key to understanding this measurement framework is understanding the availability of current metrics, and then constructing a valid model of the support process around which a measurement framework can be built.

Metrics Review

In an accompanying document (see Volume II of this work), we perform a review of existing metrics that are applicable to the software support cycle. Three classes of metrics are

outlined: *product* metrics, *life cycle process* metrics, and *behavioral* metrics.

The underlying problem with many of the proposed metrics of each of the three classes is that either they are very difficult to actually measure, or use of the metrics is not widespread. If a methodology prescribes a metric that is difficult or costly to collect, a support organization would be reluctant to follow the methodology. Therefore, our aim is to obtain the best of both worlds by selecting a subset of the proposed metrics that appear to affect the ability to support an information system and are easy to collect.

Metrics Examples

Examples of reviewed metrics which either directly or indirectly contribute to the proposed quality measures for information systems support include the following:

- Lines of Code (LOC)
- Program Age
- Module Count
- Number of Modifications
- Number of Project Personnel
- Personnel - Education Level
- Personnel - Software Engineering Experience
- Personnel - Training
- Failure Rate
- Time to Complete Maintenance Actions

In most cases, raw metrics are difficult to obtain. Many of the metrics used for building our support measures are either subjective or are simplified estimates of support characteristics. For example, while there are many proposed measures for software complexity, a subjective complexity measure may be the best one can hope for across heterogeneous environments, at least until use of a uniform complexity measurement that overcomes this hurdle becomes widespread. In addition, the exact impact of individual objective or subjective metrics on the ability to support software remains virtually unknown. The measures discussed in sections 6 through 8 are comprised of metrics believed to have the greatest impact on the value of the measures.

4 Research Objectives

Final Results

The objective of the Software Support Qualitative Assessment Methodology is the development of a set of measures (outlined in the following sections) to be used by the various

information systems support audiences. In particular, the products created to fulfill its mission are as follows:

- Supportability Measure
- Support Organization Assessment Measure
- Operational Readiness Measure

The measures are developed based on the underlying foundations discussed in sections 6, 7, and 8. Implementation information consisting of directions and recommendations for applying the measures can be found in Volumes III,IV, and V. A cost/benefit analysis detailing the advantages and disadvantages of implementing the support measures in terms of cost, effort, and other related factors can be found in Section 9.

Support Audiences

The major goal of the Software Support Qualitative Assessment Methodology is to provide software support information for a variety of support audiences. There are two major audiences: the personnel tasked with supporting information systems and the users of those systems.

Of the free measures, the first two measures are designed primarily for personnel tasked with supporting information systems. The Supportability Measure provides a focused examination of one information system. This information will be useful for the personnel working with the system as well as personnel tasked with managing the support process. The Support Organization Assessment Measure is designed for personnel tasked with managing the support process. This measure provides a encompassing view of the support organization.

The third measure, Operational Readiness, is designed primarily for system users although the current measure requires the data be gathered by the support organization. This measure provides a high level summary of the current system status. This information can be utilized by system users in deciding what systems they can rely on, and it can be used by support managers by providing comparable status information.

Characteristics of Measures

The design of these measures was guided by two desired characteristics. First, the measure should be easy to compute. And second, the measures should be easy to interpret. These characteristics provided several guidelines for the shape and feel of the measures.

That the measures should be easy to compute resulted in the following guidelines:

- Use a simple data gathering method.
- Require a minimal set of data for each measure.

- Utilize straightforward conversions from raw measures to derived measures.

Ease of interpretation implies a presentation of the measurement in a language understood by the user of the measurement. For instance, measurements may be presented as risk-based with identification of primary risk drivers. Such a measure could be easily understood by both technical and non-technical audiences. A measure may also be cost-based, which may appeal to support organization management (e.g., the cost of introducing a new support technology) . Finally, measures may be presented at a high level of abstraction, as illustrated by the operational readiness example (Section 8).

5 Information Systems Support Model

To develop measures that accurately identify and evaluate the products and aspects of the information systems support process, we must start with a representative support model. We present a model developed in [SB88]. This model contains three entities that impact the support process – the information system, the organization tasked with supporting the information system, and the group of people that use the information system. These entities are both separate and interrelated. Therefore we must be able to identify the relevant characteristics of each entity and to understand the relationships between entities.

Definition and Characteristics of Entities

Information Systems

An **information system** is composed of the collection of software that processes and produces information, the documentation that describes the operation and use of the software, and the underlying (hardware and operating system) platform. Although each component of the information system (software, documentation, platform) is vital to its proper functioning, we will concentrate primarily on the software and documentation components of the information system and less so on the underlying platform.

Historically, the majority of information systems have been batch processing systems: users would submit a "batch" of input data, the information system would process the batch of data, and the users would receive their results. Today, more real-time information systems are being built, for example, telecommunications systems. These systems are more difficult to develop and maintain, for example their interface will most likely be more complex. Information systems are typically long-lived [SB88]. Because of this characteristic, information systems are more likely to evolve over time as the number and types of people (and therefore the requirements for the system) using the system change. Finally, information systems failures are usually not life-threatening as they can be in tactical/ embedded systems, but the failures can still be quite costly and impact mission success in other ways. These characteristics, especially the last two, explain why information systems support is such an important issue.

Support Organization

An information systems **support organization** is an organized collection of procedures, personnel, and resources dedicated to support a portfolio of information systems. In most cases, information systems are not supported by the same organization (or group of people) that developed the system. Thus, the support organization personnel do not necessarily have the benefit of experience or knowledge from developing the original system. Additionally, software maintenance is often perceived as a less "glamorous" task than software development, and support groups are therefore perceived as the "step-child" of developer groups [SB88]. This perception often has an adverse effect on the support organization's ability to to maintain necessary resources and qualified personnel. The support organization must also be prepared to handle maintenance requests that may originate from a variety of sources. Not only will such requests come from a variety of different users of a system, changes may originate from other organizations, such as a federal mandate.

Users

Information system **users** are the collection of people who use the information system and its results. The user population is much more difficult to characterize than the other two support entities. This unpredictability of the user population can complicate the support process, since the type of maintenance required during the support of the system is often dependent on user requirements. What can be stated about information system users is they are usually a large, diverse group. And the support user group is often a superset of the original user group for whom the information system was originally intended. As a result, information systems in maintenance must satisfy a set of growing, changing requirements.

Measuring Support from Various Perspectives

The information system, support organization, and information system users are all important entities of the information systems support model. While these entities are interrelated, we obtain a unique perspective of support issues and problems depending upon the entity on which we choose to focus our attention. For instance, from the perspective of the information systems, factors affecting the ability to maintain the particular information system are of primary interest. From the support organization's perspective, the capability and efficiency of supporting the organization's portfolio of information systems is the primary concern. Finally, from the users' perspective, information system availability, reliability and usability are important issues. The model and the proposed measures are represented in the figure on the following page.

Therefore, in order to develop a set of support measures intended to convey a complete and accurate picture of the state of information systems support, we must accommodate each of the above three perspectives. In the following sections we discuss three high-level measures designed to address each perspective:

- Software Supportability
- Support Organization Measure

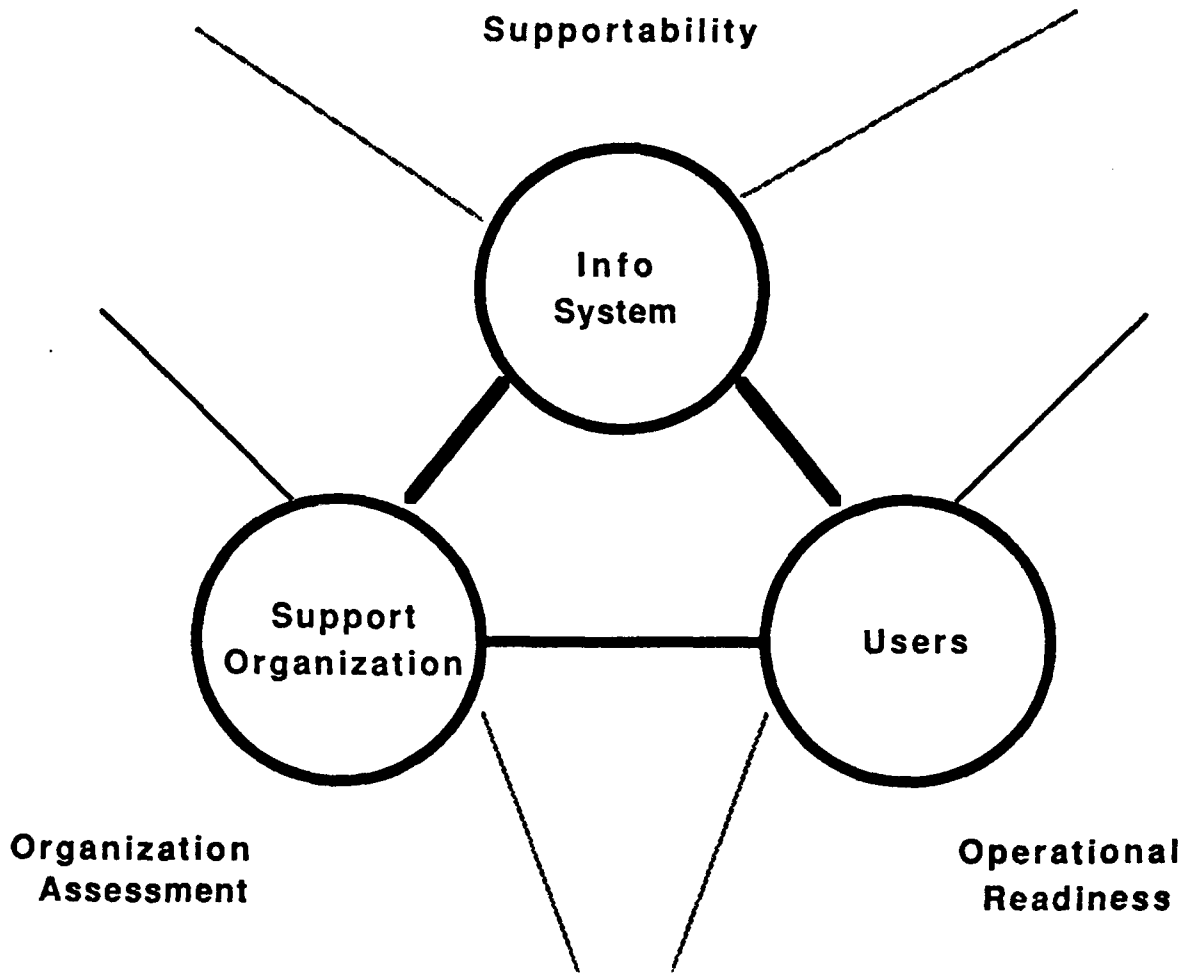


Figure 1: Quality Measures for Information Systems Support

- Operational Readiness.

6 Software Supportability

Software supportability is a measure of the effort required to satisfy user expectations of a given software product. User expectations can be divided into two groups. First, the users expect the software operation to fulfill its intended functions, i.e. its requirements. Second, users generally expect the software to be modified to meet new requirements. Factors affecting the effort required to satisfy these expectations can be divided into three categories: the software product itself, the available resources for support activities, and the management procedures used to guide the support process. More formally,

> **Software supportability** is a measure of the adequacy of products, resources, and procedures to facilitate the support activities of modifying and installing software, establishing an operational software baseline, and meeting user requirements. [PTH87]

In the following three sections, we attempt to further define the factors affecting software supportability and to break these factors down into measurable components. Following this discussion, the proposed software supportability measure is described. Implementation information for the measure can be found in Volume III.

Software Product Maintainability

The characteristics of the software product that affect the software supportability determine the maintainability of the software. Maintainability is solely a product measure. It measures the ease in which maintenance activities can be performed. Obviously, software maintenance needs to be explicitly defined before maintainability can be described further.

> **Maintenance** is all activities required to retain an item in, or restore it to, a specified condition. [Dep82]

In this case, the item is the software product which includes all programs, procedures and documentation pertaining to the operation of the system. [IEE83] Maintenance activities can be divided into three categories: *corrective*, *adaptive*, and *perfective*. Whereas *corrective* maintenance refers to changes usually triggered by a failure of the software detected during operation, *adaptive* and *perfective* maintenance refer to modifications initiated by external changes. Adaptive maintenance is initiated by changes to the operational environment; perfective maintenance is initiated by changes to the requirements. [Rom87]

Essentially, maintainability is therefore a measure of the ease with which software can be modified. Formally,

> **Maintainability** is the ability of an item, under specified conditions of use, to be retained in, or restored to, within a given period of time, a specified state

Complexity	A characteristic of the software interface which influences the resources another system will expend or commit while interfacing with the software. [CDS86]
Consistency	The extent to which uniform design techniques and notation are used. [War87]
Modularity	Characteristics which provide well-structured, highly cohesive, minimally coupled software. [War87]
Self-Descriptiveness	Characteristics which provide an explanation of the implementation of functions. [War87]
Testability	The extent to which software facilitates both the establishment of test criteria and the evaluation of the software with respect to those criteria. [IEE83]

Table 1: Design Factors Which Affect Software Maintenance

> in which it can perform its required functions, when maintenance is performed under stated conditions and while using prescribed procedures and resources. [Dep82]

Modification of software is not a trivial task. It involves such activities as program comprehension, diagnosis, repair (actually changing the software product), and testing. Many design considerations affect the ease of software modification. These factors are defined in Table 1.

Metrics for the above factors can be applied to the source code, the documentation, and possibly other parts of the software product. Other aspects of the software product can affect its maintainability. Examples include the implementation language(s) and the size of the product. It is easy to see how both of these factors could affect program comprehension.

If maintainability is viewed as a predictive measure then prediction of upcoming corrective maintenance activities is important if for no other reasons then that corrective maintenance requests will compete with adaptive and perfective maintenance requests. Obviously *correctness* is an important factor but difficult to measure. Another important factor in predicting corrective maintenance requests is the age of the software, or more directly, the extent to which the software has been previously modified. A recent study found that 83% of software faults were a result of modifications made to the software after installation. Only 17% of the faults existed in the original product. [CB87]

A summary of our proposed set of factors which we believe affect software maintainability is given in Table 2.

Software Support Management

Software support management is the collection of procedures, methods, and strategies used to direct support activities. The adequacy of the support process affects the supportability of the systems maintained under these schemes. Essentially the most efficient metrics for assessing a support process check for the existence of known software engineering techniques and subjectively evaluate their effectiveness. Example components for this factor include

Complexity	Size
Consistency	Implementation Language(s)
Modularity	Age / Number of Previous Modifications
Self-Descriptiveness	
Testability	

Table 2: Factors Which Affect Software Maintenance

the use of important standards, training of the user population, adequate forecasting of resource requirements, the ability to meet scheduled deadlines, and the employment of useful work methods.

Software Support Resources

Software support resources are made up of personnel, support systems, and facilities. The adequacy of these resources affects the supportability of the systems maintained with these resources. Personnel is composed of management, technical, support, and contractor. Support systems is composed of host, bench, lab-integrated, operational systems, configuration management systems, and other support systems. Facilities is composed of general and support facilities. [PTH87] Again, the most efficient metrics for assessing support resources check for the existence, availability, reliability, and effectiveness of the organization's resources. Example components for this factor include the training, experience, and morale of the application staff, budget constraints, existence of adequate, up-to-date software engineering tools, competing demands placed upon the application staff, the adequacy of existing hardware/software configurations, and the availability of qualified personnel.

The Software Supportability Measure

The purpose of this measure is to give the support organization a rough characterization of the supportability of an information system supported by the organization. The measure is made up three factors: system, process, and resource. The *system* factor measures components related solely to the information system. The *process* factor measures components related to the maturity and effectiveness of the process used to guide system support. The *resource* factor measures components related to the availability and effectiveness of resources critical to system support.

The measure uses two questionnaires to gather critical, consistent information about the information system and the supporting organization. Both quantitative and subjective responses are required. The construction of the questionnaire is based on a questionnaire used by Swanson to assess a variety of commercial support organizations [SB88]. The measure provides an overall rating of the supportability of an information system and specific ratings of the information system maintainability, the process under which the system is supported, and the resources which are dedicated to its support.

This process of calculating the measure consists of six steps.

- Selecting personnel to answer and administer the questionnaire.
- Reviewing the questionnaires
- Answering the questionnaires.
- Validating the questionnaires.
- Scoring the questionnaires and computing the measure.
- Interpreting the final result.

Volume III which contains the implementation details for this measure also contains guidelines for improving the supportability of an information system based on its evaluation.

7 Support Organization Assessment

The purpose of this discussion is to provide a description and explanation of the measure developed to assess the capabilities of software support organizations. This support organization assessment determines the effectiveness of the policies, procedures, resource management and personnel management of a support organization in fulfilling its objectives. We assume the total effectiveness of the organization is a sum of the organization's effectiveness in these four areas. The assessment measure described here provides a means of determining the effectiveness of an organization with regard to these four areas and a measure of determining the overall effectiveness of the organization as a whole.

The ability of an organization to support a portfolio of software applications relies on the combination of many factors. These factors are derived from characteristics of the support organization itself, the overall maintainability of the information systems being supported, and the characteristics of the users being serviced. We have collected, weighed, and organized these factors from an organizational perspective and placed them along a continuum of five levels or stages. These stages represent five levels of maturity of organizational software support capability. In order to place an organization at some point along this continuum, we have developed a set of questions that pertain to our ranked factors of organizational software support capability. The answers that are made in response to our questionnaire are combined with a formula to place an organization at a specific level of software support maturity.

We present the details of the measure by first discussing the general approach that we used in formulating our organization assessment. We describe the categories of factors that determine the effectiveness of the support organization and also describe the levels at which an organization can be classified. We then discuss our method of determining how we can place an organization at a particular level of software support capability. In Volume IV,*Implementing the Support Organization Assessment Measure*, we explain how to use our method and perform the evaluation. In that volume, we explain the evaluation scheme, the method of determining a score and the method of interpreting the score. We conclude this discussion by emphasizing the role of this organizational assessment within the context of the total software support qualitative assessment process.

General Approach

The importance of evaluating support organizations follows from the fact that such a large percentage of the software life cycle is devoted to support. It has been estimated that support costs consume more that 70% of the total life cycle cost of software development [SB88]. Instead of support costs decreasing, Swanson found that the costs of maintaining a given software package increase over time. Even with increasing use of structured techniques both in the design of new systems and in retrofitting older systems, the costs of software support are still high. With such a great proportion of the life cycle costs incurred for software support, it is very important to understand and measure the process of software support in order to reduce these costs. The primary goal of the measure is to help a support organization evaluate its support capabilities.

Although we have modeled our measure and its development on the work done by Humphrey [HSE+87] and the Software Engineering Institute (SEI), there is an important distinction: Humphrey's SEI methodology assesses organizations tasked with *developing* software systems whereas our measure assesses an organization's software *support capability.* Although from a traditional viewpoint, software support is considered a subset of the total software development process, there are several reasons for concentrating solely on software support. First, because of the high costs incurred with software support, it cannot continue to be treated as an afterthought of the development process. Second, many organizations are solely software support organizations and do not perform software development. Third, the emphasis of the software support process is different than the emphasis of the development process: The emphasis in software development is on problem analysis and requirements definition and design. In software support, the emphasis is on problem analysis, systems analysis of the existing system, and expedient problem resolution.

For these reasons, we feel it is important to evaluate maintenance organizations separately from development organizations. Many factors that determine quality software development also determine quality software support. But these factors have different weights depending upon whether we are measuring an organization's ability to develop software or provide support. The focus of these evaluations is critically different. And perhaps the greater contribution is to be made in the software support arena.

Factors Influencing Software Support Capability

We have categorized the factors, or issues, into four groups: organizational issues, software support process management, tools and technology, and personnel.

Organizational Issues

Organizational issues deal with organizational policies and procedures. The factors include:

1. **the structure of the organization**: issues relating to how the groups within the support organization are organized, reporting and control structure, span of control, formal job descriptions of personnel, composition of software teams, etc.

2. **the characteristics and management of the portfolio of applications that is being supported**: issues that concern age, size, languages of the application portfolio, development background, consistency and standardization across applications, and documentation issues.

3. **the physical environment**: this includes access to systems and resources for emulating user environments as well as access to resources for performing required changes to the software systems.

4. **budgetary control**: relative size of budget and control measures,

5. **organizational effectiveness measures**: how the organization perceives how it is measured by the parent organization, and

6. **relationships with the development and user organizations**: user literacy, communication with users and developers, frequency of communication, negotiation channels, and user expectations.

Software Support Process Management

Software support process management factors also deal with policies and procedures but these policies and procedures of concern here concentrate on factors such as process metrics, standards, and the management mechanisms that are used in managing the software support process itself. It also involves an understanding of the types of problems that the organization must be expected to undertake. For example, software support activity pertaining to one application can be classified as performing corrective maintenance, adaptive maintenance, or perfective maintenance [SB88].

Factors can be grouped into three main areas:

1. **standards and procedures**: policies and rules that pertain to how the organization maintains each information system,

2. **process metrics**: the measures used for assessing performance of the maintenance task, and

3. **management of the support process**: policies, procedures, and mechanisms that the organization uses to manage the complete application portfolio rather than each individual application.

Tools and Technology

Tools and technology along with personnel factors assess an organization's ability to use the resources available to the organization effectively. Issues concerning tools and technology include:

1. **technology management**: understanding existing maintenance technology in the industry,

2. **use of tools in the support process**: what tools and techniques are actually used by the organization,

3. **tools management**: particularly with respect to software development, and

4. **documentation tools**: which, if any, tools are used.

Personnel Issues

Personnel issues are an important set of factors which affect the support capability directly as well as indirectly. Personnel training and experience have a direct impact which can be readily measured. Issues such as formal training methods and job rotation of experienced employees come under this category. Employee turnover rate, recruitment procedures, and motivation levels form a good set of indicators to identify potential problems. These indicators along with factors such as manager and staff relationship have an indirect impact on support quality.

Levels of Software Support Capability

Based on Humphrey's Maturity Framework [HSE+87] we posit the following five levels of software support capability for classifying support organizations: Ad-hoc, Repeatable, Methodology, Control, and Optimal. A detailed description of each level along with possible symptoms that can help identify each level are provided below.

1. **Ad-Hoc**:
   Organizations that maintain software at the ad-hoc level manage in a chaotic manner. There are no formalized procedures for support. Technology and tools are not modern, not fully understood, nor properly integrated within the software support process. Change control is lax and senior management is inexperienced with little understanding of problems and issues resulting in delays and high costs. Some of the possible symptoms of this level are:

   (a) low morale/motivation among staff

   (b) inexperienced users and lack of understanding of the system

   (c) absence of adequate technology or methodology

   (d) no emphasis on documentation or measurement of performance

   (e) no quality assurance and lack of upper management involvement in operations

   (f) lack of communication among staff, with users or developers

   (g) lack of formalized training/user support procedures

2. **Repeatable**:
   In the repeatable phase, the organization has mastered the repetitive support processes, however it is unable to face new challenges. The organization uses standard methods and practices for software support activities such as problem recording and classification procedures, code changes, requirement changes, etc. The bulk of the

support activity of an organization performing at this level is corrective maintenance and adaptive maintenance with little perfective maintenance being performed. The symptoms include:

(a) growing understanding of support issues, but poor planning

(b) concern for better system control and measurements

(c) basic understanding of maintenance problems but solutions tend to be quick fixes

(d) there are efforts to improve communication

(e) requirements specifications exist

(f) unable to undertake challenging assignments

3. **Methods**:
At this level, the software support process is well understood and well defined. This allows for consistent implementation. There is a well-defined support philosophy, a set of concepts and principles which governs the support function. However, there is no feedback mechanism in the system to measure the performance of the support functions with a view to improve the effectiveness of the process. In other words, while the concepts and principles are well-defined and documented, there is no evidence that these guidelines are actually followed. The symptoms include:

(a) improving perception of maintenance role by users and improved knowledge of applications

(b) emphasis on documentation/source code, modularity, consistency issues

(c) emphasis on reducing maintenance efforts by improving software quality

(d) systematic, defined support procedures

(e) adequate user support and training

(f) existence of formal change requests and good communication

4. **Control**:
At this level, measurements exist to indicate that the concepts and principles of the organization support philosophy are actually being applied. Whereas an organization operating at the methods level has specified the concepts and principles, an organization operating at the control level actually can actually demonstrate that these concepts and principles are applied to the support process. Indicators that an organization is operating at this level may include:

(a) substantial quality improvements in the jobs that are being done

(b) an increasing amount of perfective maintenance being performed on each application

(c) systematic and periodic check-ups of each application

(d) formally documented change control records

(e) focus on improving the support quality by concentrating on measuring elements of the support process

(f) data is gathered and measurements for support products and tools are recorded

(g) use of evaluation methods for tools, techniques and products which are used for improving the support activity

5. **Optimal**:
   At this level, support organizations have not only achieved a high degree of control over their process, they have a major focus on improving and optimizing their operations. The support function is well organized within each area of application. Support training is an integral part of the functions. The maintenance function is perfective in the sense that it is performed to eliminate processing inefficiencies, enhance performance, and improve maintainability. There is sophisticated analyses of the error and cost data and prevention mechanisms for such errors. The symptoms at this level are:

   (a) clear cut perception of software support function and application portfolio by the users

   (b) well maintained application systems portfolio with specific measures of product support and quality

   (c) application of process control measures and obtaining improvements in support function as a result

   (d) well managed procedures for training and user support

   (e) good communication with well laid out formal procedures for change requests and maintenance of all types

   (f) disciplined environment frees the talented staff to be creative instead of solving crises

These five levels of Software Support Capability represent levels of maturity for software support management (Figure 2). These five levels represent a path of knowledge and practices that reflect the ability of an organization to manage the software support process.

Summarizing the Support Organization Assessment Measure

The process of software development is an evolving process. Better methods and procedures are still being defined. But even with this evolving process it is possible to measure an organization's maturity with respect to how it performs this process [Hum89].

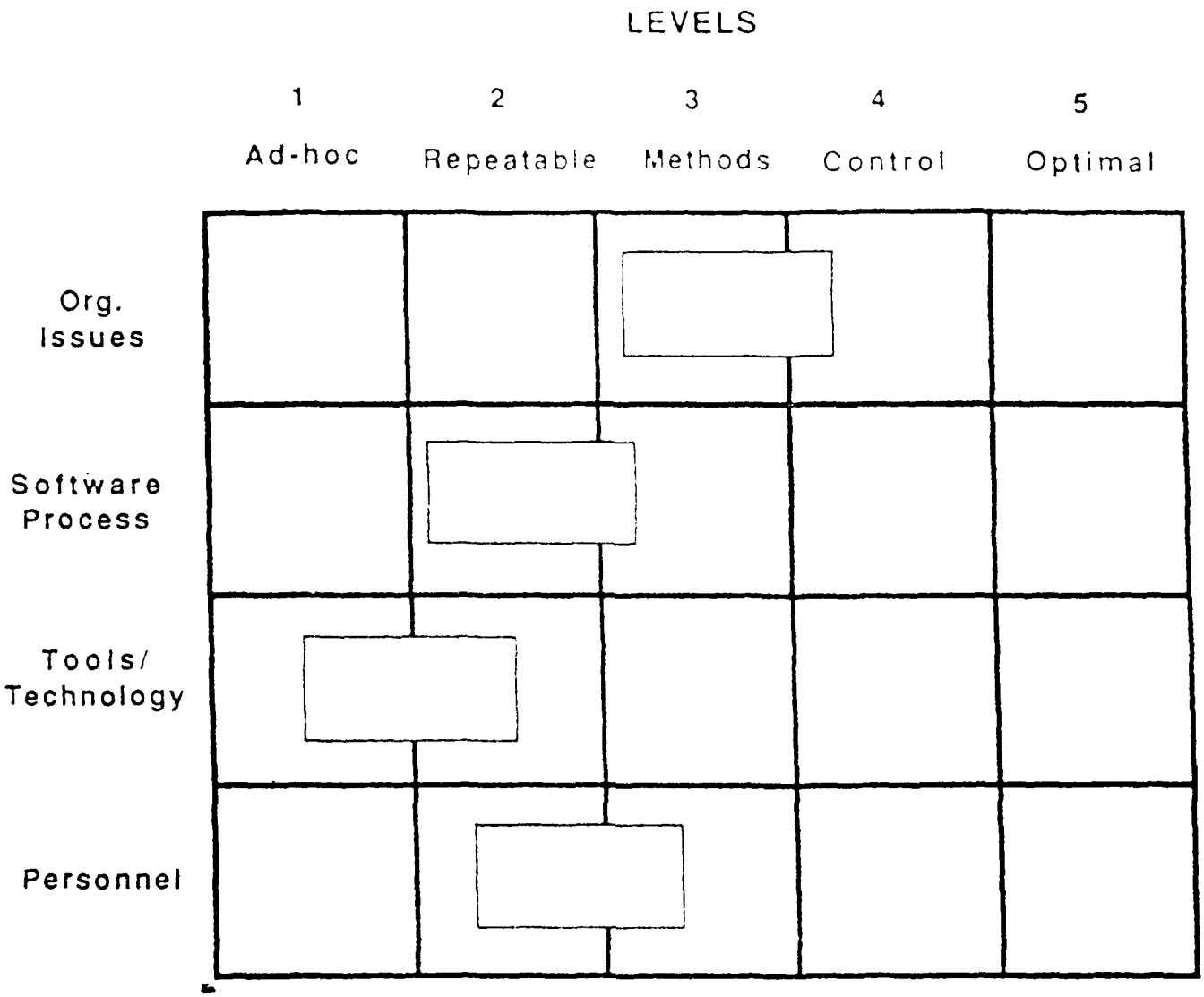
The process of software support is less structured and less understood than the larger process of software development. As such, we expect the factors we have enumerated in our measure will change in content and importance as more information is gathered with respect to how organizations perform software support. Nevertheless, it is still possible to measure how well an organization understands and manages the task that it is chartered to perform. The measure of a software support organization depends upon how well an organization understands software support and how well it manages the software support process.

We have used the SEI Model to develop this support organization assessment measure. We used Swanson and Beath's information system support model to determine an organization's position in the information system and listed all of the factors pertaining to

software support from the organizational point of view in the information systems model. We grouped these factors into four major categories and placed them upon a maturity matrix. By answering "yes" to questions that probe these factors we feel we can place an organization at a point on the maturity level based upon collapsing the maturity matrix onto a line.

This method of organizational assessment is not intended to be an overall evaluation of the organization. Certain aspects of software support are outside the control of the organization. The organization may not have any choice in the applications contained within its portfolio of supported supported systems.

Also, the ultimate assessment of the software support organization may result from the users of the supported systems. We might try to assume an organization operating at Level 5 will have cooperative and enthusiastic users, but this cannot be guaranteed. To this extent, one must exercise caution in reading the results of the evaluation using our questionnaire. The results need to be tempered by the considerations outlined above.



Hypothetical Organization Ranking

FIGURE 2

8 Operational Readiness

Operational readiness is another measure an organization may use to gauge its effectiveness in fulfilling its support task. It is also a useful measure for the users of an information system. The **operational readiness** of a software system is the the ability of the software system to effectively perform its intended function based on the following:

- The current operational state of the system,
- The reliability of the system, and
- The supportability of the system.

Operational readiness addresses such questions as, "Will the system be up and running when I need it?", and, "When I use the system, can I expect correct results?" Our view of operational readiness is that it is mainly a predictive measure. The assessment of a software system's state of operation in a present or past tense is a trivial problem – either the system is operating correctly or it is not. However, a more useful and much more difficult problem to solve is the determination of whether an information system will successfully "complete its mission" at some point in time in the near future.

Characteristics of Operational Readiness

Like supportability, operational readiness is a risk-based measure. Whatever metric units chosen for representing operational readiness, operational readiness is essentially a measure of the probability that software will perform its intended function. We must take into account expectations of software performance and maintenance activity (from the user's and supporter's perspective, respectively) along with the actual values of these two items. The impact to a user of a particular failure will affect the importance of the parameters associated with the appearance of such a failure and the resultant risk. Likewise, risk will be partially determined by the adequacy of support management's planning for maintenance activities. Because the results of this research are intended for use by support organizations, our interpretation of operation readiness is biased towards measuring characteristics obtainable in a support organization environment. In the future, we hope to additionally study user organizations and improve the existing measure.

A unique characteristic of operational readiness is that it is more subject to random variations in the information systems support environment. The amount and type of information system maintenance requests and the maintenance repair schedule are constantly changing. Thus, while many of the elements of operational readiness are also elements of supportability, the operational readiness is more likely to alert a support organization to potentially significant short-term problems and allow the organization to effectively respond to the problems.

Although there are many possible units in which operational readiness may be expressed, we borrow the terminology that has been applied to military equipment. Three terms are used to denote operational readiness: red, yellow, and green. These terms indicate one of three basic "states" of readiness. A state of **red** indicates the system is in a serious state

of disrepair. A state of **yellow** denotes caution – the system can still perform as intended, but pending difficulties may cause the state to deteriorate to red unless the difficulties are solved. A state of **green** indicates that the information system is "healthy" and is functioning without difficulty.

Operational Readiness Components

When measuring the operational readiness of an information system, we want to identify those characteristics impacting the ability of users to operate the system as intended when needed. Some of the characteristics describe the ability of the users to effectively operate the system, while others identify the "state of maintenance" of the information system. The ability of the users affects operational readiness, since misdiagnosed failures and improper maintenance requests can originate from an ill-trained, inexperienced user group. The "state of maintenance" of an information system, a term describing the backlog of system maintenance requests and associated information, can affect operational readiness depending upon the type and urgency of pending requests and the time required to complete those requests. In addition, a high-level measure of the overall support of the given information system is an important factor, since the support organization and information system itself can impact operational readiness irrespective of the other characteristics.

The list of operational readiness components are as follows:

- Current state of information systems maintenance
  - Support staff availability
  - Volume of pending maintenance requests
  - Maintenance repair schedule difficulties
  - Number and rate of system failures
- System reliability
  - Proportion of corrective maintenance requests
  - Proportion of emergency maintenance requests
  - Amount of system "down" time
- System Supportability
  - System Size (Lines of Code)
  - Language(s)
  - Average source code module size
  - System age / length of support
  - Total number of modifications
  - Documentation availability
  - Documentation adequacy
  - Personnel capability
  - Software/hardware platform adequacy

Implementing an Operational Readiness Measure

Guidelines for implementing an operational readiness measure as part of a set of information systems support quality measures are given in Volume V of this work. As indicated in the previous section, the operational readiness measure consists of three main factors: the "current state" information, reliability, and supportability.

The measure utilizes a questionnaire to gather a mix of subjective and objective data on an information system and the state of maintaining the system. The process of calculating the operational readiness measure is similar to that of calculating the supportability of an information system (Section 6).

9 Cost – Benefit Analysis

One goal of the Software Support Qualitative Assessment Methodology is the development of measures that would not be costly to collect and that would benefit an information systems support organization by providing a foundation for the improvement of their support process and a reduction of the support cost.

In the following sections, we outline the expected cost of implementation of the methodology in terms of materials expended, personnel involved, and time required. In addition we outline the benefits, which are expected to outweigh any incurred costs.

Materials and Resources

There is a minimum of materials required to implement the support measurement program within a support organization. The required materials to collect the three measures of support organization assessment, supportability, and operational readiness are located in Volumes III, IV, and V of the methodology.

No additional resources are required to implement the methodology itself. We expect, in the future, this methodology will be implemented via an automated process. The required presence of resources to supplement the automation of the measure collection and calculation process would be outweighed by savings in time required to gather data (see below).

Personnel

The cost of implementing this methodology in terms of personnel depends to some extent upon the personnel selected to collect the measures. As mentioned in the guidelines for implementing the measures (Volumes III-V), the selection of appropriate personnel to complete and analyze the questionnaire is crucial to the successful implementation of the methodology.

Aside from the issue of appropriateness, the number of personnel required to implement the methodology depends upon the number of information systems supported and the number of people supporting the information systems. As mentioned in the guidelines for

implementing the measures, the more qualified personnel available to complete the questionnaires, the more accurate the measures are likely to be. At the least, 2 people per support organization should complete and analyze an organization questionnaire, and 2 people per information system should complete and analyze a system questionnaire.

Time

The amount of time required to carry out the methodology is dependent on the availability of easily accessible system data and the number of personnel tasked to complete the questionnaires. As a general rule, the amount of time required to complete the organization questionnaires will vary from 4 person-hours to 24 person-hours, depending upon the availability of existing organization information, the size of the support organization, and the number of personnel completing the questionnaires. The amount of time required to complete the system questionnaire will vary from 4 person-hours to 12 person-hours depending upon the availability of system data.

Benefits

We expect the benefits of implementing our methodology will easily outweigh the costs involved. The exact quantification of benefits are thus far undetermined, as additional studies to validate the measures are necessary. However, the most important benefits likely to be gained are as follows:

- Provision of insight into support process
- Provision of a foundation for sustained improvement of support process
- Estimation of the impact of changing support resource allocations or procedure plans
- Justification of resource and/or procedure changes

The most important benefit to be gained is the provision of insight into the support process. The understanding of a process begins with measurement, and the Software Support Qualitative Assessment Methodology provides a complete but not overly exhaustive set of measures. The support measures also provide a foundation upon which, depending upon subsequent actions, a sustained improvement of the support process can occur. The implementation of the measures also improves the capability of the support organization to gauge the impact of changing or introducing resources and procedures and to justify such changes.

The exact quantification of benefits are thus far undetermined, as additional studies to validate the measures are necessary.

10 Future Research

The goal of this research has been the development of high-level information systems support measures via a state-of-the-art metrics review and a case study conducted by CIMR of U. S. Army information systems support organizations. The result of these efforts are three high-level support measures comprised of certain key factors believed to most heavily influence different perspectives of information systems support. The results also contain the initial support qualitative methodology and methodology implementation guidelines. This study lays the foundation for additional software support studies designed to refine and validate the qualitative assessment methodology. In addition, the results of this project are useful in the study of reverse engineering, a process closely coupled with the support of software systems.

Additional Support Studies

Whereas the results of this initial study included the initial development of a software support qualitative assessment methodology, a continuing study would involve a more thorough evaluation of the initial findings. The study would involve selecting a subset of field study factors appearing to have the greatest influence on the ability to support an information system and conducting a statistical validation of these factors. The selection and validation process would, in turn, lead to a refining of the initially proposed methodology. The refined methodology can then be implemented in a selected setting and the implementation results can be observed.

This research serves as the foundation for other studies as well:

- Testing the refined software assessment methodology in several controlled settings, such that one or more kay parameters (such as support organization size) is varied.
- The development of tools for support management and staff to use to carry out the support assessment methodology.
- A more focused study of information system users. The study would specifically focus on user needs and problems and the (often weak) interface between users and supporters of information systems.

Reverse Engineering Studies

Reverse engineering is the part of the maintenance process that helps in understanding the software application [CC90]. Reverse engineering can be a valuable aid in comprehending a program, especially if the documentation for a program is incomplete, incorrect, or nonexistent. In addition to serving as a simple program comprehension tool, reverse engineering can help retrace the translation from design to source code such that a software system can be reprogrammed.

To date, no known study has identified factors critical to the decision to reverse engineer a software product. Obviously, such a decision is made in the support environment.

Therefore, many of the factors influencing the ability to support an information system likely affect the decision to reverse engineer as well. This current research is a natural prerequisite for reverse engineering decision studies.

Possible studies of reverse engineering decision making include the following:

- The development of a reverse engineering decision model based on factors identified through empirical observation.
- A study of state-of-the-practice reverse engineering methodologies and the development of a more general reverse engineering methodology based on the study.
- Development and refinement of models to estimate cost and risk factors associated with reverse engineering.

11 Conclusion

The Software Support Qualitative Assessment Methodology is based on the premise that a single high-level software support measure may not accommodate all viewpoints. The support organization is most likely primarily concerned with its ability to support its portfolio of software systems, users are more concerned with the "operational readiness" of a software system, and the support technicians are concerned with product supportability. In addition, injecting quality measures for information systems support is expected to lead to greater understanding of the support process. This greater understanding, in turn, serves as a foundation for improving the support process, reducing support cost, and improving support efficiency.

This research recognizes that the support process is currently ill-defined and additional studies are required to analyze the information systems support environment and to refine the proposed support measures. Our goal is to equip information systems supporters and users with the appropriate knowledge base and tools to analyze support issues for themselves and possibly applying the results of this research to other phases of the software engineering life cycle.

A Glossary of Terms

Acceptance Review A review of a software product by developers and maintainers to determine if the product satisfies all originally specified requirements.

Acceptance Test Testing led by the client or QA group to determine whether the product satisfies its specifications as claimed by the developer.[Sch90]

Application System same as Information System

Availability A measure of the degree to which an item is in an operable and committable state at the start of a mission when the mission is called for at a random point in time.[Dep82]

Benchmark Testing Evaluation of the system performance against quantitative requirements.[Sch90]

Change Request Review Board An authority responsible for evaluating and approving requests for changes to a software product.

Cohesion A measure of the degree of the functional relatedness within program units. [Som89]

Complexity A characteristic of the software interface which influences the resources another system will expend or commit while interfacing with the software. [CDS86]

Configuration Management The process of identifying and defining the configuration items (hardware/software units) in a system, controlling the release and change of these items throughout the system life cycle, recording and reporting the status of configuration items and change requests, and verifying the completeness and correctness of configuration items.[IEE83]

Consistency The extent to which uniform design techniques and notation are used. [War87]

Coupling A measure of the strength of interconnections (dependencies) between program units. [Som89]

Error Human action that results in software containing a fault. Examples include omission or misinterpretation of user requirements in a software specification, incorrect translation or omission of a requirement in the design specification. [IEE83]

Failure A departure of program operation from program requirements.[IEE83]

Failure Rate The number of failures of an item per measure-of-life unit.[Dep82]

Fault A manifestation of an error in software. A fault, if encountered, may cause a failure. Synonymous with bug.

Fourth Generation Language (4GL) A computer programming language that provides abstractions of data and/or procedural specifications and is usually suited for a particular application domain.

Integration Testing Verify that the modules of the system combine correctly in order to achieve a product that meets its specifications. [Sch90]

IS (Information Systems) Organization An organied collection of procedures, personnel, and resources dedicated to support a portfolio of information systems.

Lines of Code Lines of source code, not including comments.

Maintainability The probability that an item will be retained in, or restored to, a specified condition within a given period if prescribed procedures and resources are used.[Dep82]

Maintenance All actions required to retain an item in, or restore it to, a specified condition.[Dep82]

Maintenance Audit An organized review of the maintenance organization.

Maintenance Escort Participation of the software maintainer in software system development.

Man/Machine Interface The software that supports the interaction between the user and the system.

Measure A high-level unit of specification which characterizes, evaluates, or predicts various aspects of software life cycle processes and products.

Metric A measurable indication of some aspect of a system. [DeM82] A quantification of a specific feature of the software life cycle process or software product.

Modularity A characteristic of software such that it is well-structured, highly cohesive, and minimally coupled. [War87]

New Systems Development The development of a system which has never been fielded.

Object Oriented Design Designing a system in terms of abstract data types where the objects are instantiations of the data types and new data types can be defines as extensions of previously defined types.

Regression Testing Testing the system against previous test cases to ensure that the functionality of the system has not been compromised by recent changes to the system. [Sch90]

Reliability The probability that an item will perform its intended function for a specified interval under stated conditions.[Dep82]

Self-Descriptiveness A characteristic of software that enables the understanding of implementation of software functions. [War87]

Support Staff The personnel tasked with maintaining an information system.

Supportability A measure of the adequacy of products, resources, and procedures to facilitate the support activities of modifying and installing software, establishing an operational software baseline, and meeting user requirements. [PTH87]

Testability The extent to which software facilitates both the establishment of test criteria and the evaluation of the software with respect to those criteria. [IEE83]

Throw-away prototyping Creating a prototype as part of system design and then "throwing away" the prototype and implementing the system "from scratch" not using any of the source code from the prototype.

Top-down design Designing the system by recursively breaking the system down into smaller components.

Unit Testing Testing of individual portions of the system.

B List of Acronyms

AIRMICS U.S. Army Institute for Research in Management Information, Communications, and Computer Science

AMC Army Materiel Command

CCB Change Control Board

COE Army Corps of Engineers

FORSCOM Forces Command

HSC Army Health Services Command

IS Information System

ISC Army Information Systems Command

LOC Lines of Code

C Summary of Sites Contacted

Health Services Command (HSC)

Key Personnel

Dee Lawrence	512-471-4475	Health Care Systems Support Activity - Ft. Sam Houston
Ralph Coogan	512-471-4475	

Summary of Involvement

U. S. Army Health Care Systems Support Activity (HCSSA) at Fort Sam Houston, Texas, agreed to participate in study. Their site served as an excellent testbed for the refinement of the initially developed support model. HCSSA contributed information for 3 support organizations and 17 information system. They expressed interest in possible follow-on studies.

Systems Surveyed

Burroughs Computerized Appointment System
Area Dental Lab System
The Army Auth Document System
HSC Local Force Development System
HSC Local Finance and Accounting System
Comptroller Management Indicator System
Med Customer Auto Support Package System
Med Stock Control System
Extension Service Div System
Scheduling System
Health Risk Appraisal System
Individual Patient Data System
AMEDD Property Accounting System
Theater Army Medical Management Information System
Uniform Chart of Accounts Personal Utility System
Expense Assignment System II
Workload Management System for Nursing

Information Systems Command (ISC)

Key Personnel

Janet O'Keeffe	703-355-7098	ISSC - Technical Support Directorate
Lt. Col. Kerrigan	703-355-7166	
Ival Secrest		SDC - Ft. Huachuca
Kathy Moyers	317-542-3352	SDC - Ft. Benjamin Harrison
Arlene Aldridge	804-734-1450	SDC - Ft. Lee

Summary of Involvement

Although ISC never formally declined to participate in this study, no opportunities to gather data at any ISC facility arose.

Forces Command (FORSCOM)

Key Personnel

Melba Jackson	404-669-5707
Casby Harrison	404-669-5786

Summary of Involvement

FORSCOM provided available support organization data. Because of special circumstances that arose during the course of the project, FORSCOM was unable to contribute a full set of information regarding their portfolio of information systems.

Army Materiel Command (AMC)

Key Personnel

George Sumrall	201-544-4273	AMC Headquarters
Ray Mosman	314-263-5045	Systems Integration and Management Activity - St. Louis
Claude Williams	314-263-5884	
Robert Marshak	314-263-5978	

Summary of Involvement

A visit to the Systems Integration and Management Activity (SIMA) yielded data for one organization and one very large information system.

Systems Surveyed

Commodity Command Standard Subsystem

Corps of Engineers (COE)

Key Personnel

Jim Johnston 203-653-1248

Summary of Involvement

COE declined to participate in this study.

References

[CB87] J. S. Collofello and J. J. Buck. Software quality assurance for maintenance. *IEEE Software*, pages 46–51, September 1987.

[CC90] E. J. Chikofsky and J. H. Cross. Reverse engineering and design recovery: A taxonomy. *IEEE Software*, pages 13–17, January 1990.

[CDS86] S. D. Conte, H. E. Dunsmore, and V. Y. Shen. *Software Engineering Metrics and Models.* Benjamin/Cummings, Menlo Park, CA, 1986.

[DeM82] T. DeMarco. *Controlling Software Projects.* Yourdon, New York, 1982.

[Dep82] Department of Defense. *Test and Evaluation of System Reliability, Availability, and Maintainability: a Primer*, March 1982. DoD Directive 3235.1-H.

[HSE+87] W. S. Humphrey, W. L. Sweet, R. K. Edwards, G. R. laCroix, M. F. Owens, and H. P. Schulz. A method for assessing the software engineering capability of contractors. Technical Report CMU/SEI-87-TR-23, Software Engineering Institute, September 1987.

[Hum89] W. S. Humphrey. *Managing the Software Process.* Addison-Wesley, Reading, Mass., 1989.

[IEE83] IEEE. *IEEE Standard Glossary of Software Engineering Terminology*, 1983. Standard 729-1983.

[PTH87] D. E. Peercy, E. Tomlin, and G. Horlbeck. Assessing software supportability risk: A minitutorial. In *Conference on Software Maintenance*, pages 72–80. IEEE, 1987.

[Rom87] H. D. Rombach. A controlled experiment on the impact of software structure on maintainability. *IEEE Transactions on Software Engineering*, SE-13(3):344–354, March 1987.

[SB88] E. B. Swanson and C. M. Beath. *Maintaining Information Systems in Organizations.* John Wiley and Sons, New York, 1988.

[Sch90] Stephen R. Schach. *Software Engineering.* Aksen Associates Incorporated Publishers, Boston, MA, 1990.

[Som89] Ian Sommerville. *Software Engineering.* Addison-Wesley Publishing Company, New York, 1989.

[War87] J. L. Warthman. Software quality measurement demonstration project (i). Technical Report RADC-TR-87-247, Rome Air Development Center, December 1987.